

Apache Flink Word Count App

User

2025-01-17, 06:50:06

Generate a Apache Flink application using Scala language to solve the problem! Given a plain text UTF-8 file URL and a output CSV file URL, write in the output file the occurrence sum of each word in the input file.

Assistant Llama 3.1 70b instruct

2025-01-17, 06:50:06

Below is a basic example of how you can achieve this using Apache Flink in Scala. This application reads a plain text UTF-8 file, counts the occurrences of each word, and writes the result to a CSV file. Please ensure you have the necessary dependencies in your `build.sbt` file for Apache Flink, including the Flink Scala API and the necessary connectors for file systems.

```
```scala
import org.apache.flink.api.common.functions.{FlatMapFunction, ReduceFunction}
import org.apache.flink.api.common.typeinfo.Types
import org.apache.flink.api.java.tuple.Tuple2
import org.apache.flink.streaming.api.scala.{DataStream, StreamExecutionEnvironment}
import org.apache.flink.util.Collector
object WordCount {
 def main(args: Array[String]): Unit = {
 // Set up the execution environment
 val env = StreamExecutionEnvironment.getExecutionEnvironment

 // Read the input file
 val text: DataStream[String] = env.readTextFile("path_to_your_input_file.txt")

 // Process the data
 val counts: DataStream[(String, Int)] = text.flatMap(new Tokenizer).keyBy(0).reduce(new SumReducer)

 // Write the data to the output CSV file
 counts.writeAsCsv("path_to_your_output_file.csv")

 // Execute the program
 env.execute("Word Count")
 }

 class Tokenizer extends FlatMapFunction[String, (String, Int)] {
 override def flatMap(value: String, out: Collector[(String, Int)]): Unit = {
 // Split the line by spaces and emit each word with a count of 1
 value.toLowerCase.split("\\s+").foreach { word => out.collect((word, 1)) }
 }
 }

 class SumReducer extends ReduceFunction[(String, Int)] {
 override def reduce(value1: (String, Int), value2: (String, Int)): (String, Int) = {
 // Reduce two values of type (String, Int) into one (value1._1, value1._2 + value2._2)
 (value1._1, value1._2 + value2._2)
 }
 }
}
```
```

This example will work for a batch-processing scenario where the input file is static and can be fully loaded into memory. For larger datasets or real-time data streams, consider using Flink's `DataStream` API with more robust input sources and handling mechanisms. Remember to replace `"path_to_your_input_file.txt"` and `"path_to_your_output_file.csv"` with the actual paths to your files. This application reads a text file line by line (`readTextFile`), splits each line into words (`Tokenizer`), sums up the occurrences of each word (`SumReducer`), and writes the result to a CSV file (`writeAsCsv`). The Flink configuration, the execution environment, and the data loading and processing steps are all encapsulated within the `WordCount` object. The result will be a CSV file where each row contains a word from the input file and its count of occurrences, separated by a comma.